Poner en marcha un servidor en casa con DNAT y DNS dinámico

Si montamos un servidor cualquiera en una máquina con IP privada, solo admitirá en principio conexiones de clientes pertenecientes a su misma red local. ¿Cómo podríamos hacer entonces que nuestro servidor estuviera disponible públicamente para poder recibir peticiones desde cualquier cliente del mundo?

Solución profesional: IaaS

Una posible solución es alquilar un servicio de tipo IaaS. Este tipo de servicios consisten en disponer de una máquina totalmente personalizable (a nivel tanto de hardware -nºcpus, memoria, etc- como de sistema operativo -a elegir entre varias distribuciones de Linux o más-) que estará alojada en un servidor remoto (propiedad de la empresa que ofrece el servicio) pero que tendrá una determinada IP pública (y por tanto, accesible directamente desde Internet) y que podremos gestionar como administrador (normalmente a través de SSH), teniendo, pues, el control absoluto sobre ella (para, por ejemplo, poder instalar y configurar el software que deseemos). Ejemplos de empresas que ofrecen servicios IaaS son: Linode (https://www.linode.com), DigitalOcean (https://www.digitalocean.com), OVH (https://www.ovh.es) o ScaleWay (https://www.scaleway.com), entre muchas otras. No profundizaremos aquí en ellas porque se necesita desembolsar dinero.

Solución casera: DNAT

Otra posible solución que, a diferencia de la anterior, es gratuita (si no tenemos en cuenta el pago de nuestro acceso estándar a Internet) es "vincular" al servidor en cuestión de nuestra red local la IP pública de la puerta de enlace (es decir, del "router" de nuestra red), para que cuando esta reciba un mensaje se lo reenvíe a aquél (y viceversa). Para ello:

1.- Debemos entrar en el panel de control del "router"; la manera de conseguir esto varía según cada modelo pero normalmente el acceso se realiza vía web especificando en un navegador la IP privada del "router" y, en el cuadro entonces mostrado, un nombre de usuario y contraseña. Estos tres datos los ha de proporcionar nuestro operador telefónico.

> NOTA: La IP privada del router la podemos conocer, de todas maneras, simplemente observando la salida del comando *ip route show* Por otro lado, los nombres de usuario y contraseñas por defecto más habituales en los modelos de "routers" domésticos más comunes se pueden encontrar en listados publicados en Internet, como por ejemplo en http://www.adslayuda.com/password_router.html

2.-Una vez dentro del panel de control, debemos encontrar una sección llamada habitualmente "(D)NAT" -"(Destination) Network Address Translation"- (o "Port Forwarding") desde donde podremos definir esta "vinculación". La manera concreta vuelve a variar otra vez según el modelo de "router", pero en esencia lo que debemos hacer allí es una redirección de puertos para que los mensajes que lleguen desde el exterior a un puerto específico del "router" (accesible gracias a su IP pública) sean reenviados a otro puerto específico de nuestro servidor (identificado por su IP privada). Es decir, si nuestro computador (con IP privada 192.168.1.111, por ejemplo) tuviera el puerto 80 abierto porque tiene funcionando un servidor web, tendríamos que indicar que todo lo recibido por el puerto 80 del "router" fuera redirigido al puerto 80 del computador 192.168.1.111. Haciendo esto ya tendríamos nuestro servidor listo para escuchar peticiones de todo el mundo, literalmente (hay que tener en cuenta, no obstante, que cada puerto del router sólo puede ser redireccionado una vez).

> NOTA: Hay que tener en cuenta un detalle más, no obstante: por defecto, la mayor parte de los "routers" incorporan un servidor DHCP embebido mediante el cual asignan IP locales a todos los elementos que se conectan a la red privada. Cada cierto tiempo (cuando se termina la "concesión") esas IP asignadas pueden cambiar, por lo que la IP privada concreta del servidor indicada en el apartado de "Port Forwarding" explicado en el párrafo anterior puede ser que deje de ser válida y, por tanto, haya que modificarla convenientemente. Para que esto no suceda, se le puede indicar al "router" que no modifique nunca la IP asignada inicialmente a nuestro servidor local. Para ello, se debe conocer la dirección MAC de dicho servidor porque, en definitiva, de lo que se trata es de indicarle al "router" que a esa dirección MAC se le asigne siempre la misma IP. Concretamente, se ha de acceder a las propiedades del servidor DHCP embebido a través de los menús del "router", (suele haber alguna sección llamada "DHCP" o "DHCP Settings" situada bajo el menú "LAN" o similar). Una vez allí se pueden ver cosas interesantes como el rango de IP que asigna el servidor DHCP (esta opción puede que venga como “IP Pool Range” o como direcciones IP inicial y final), etc pero lo que en este caso se debe buscar es alguna opción llamada “Reserva de direcciones IP” o “MAC-Base Assignment” o similar, que será donde se pueda asociar la MAC del servidor local con la IP concreta que deseemos asignarle de forma fija.

NOTA: Es posible que, aún realizado el "Port-Forwarding" y la asignación de IPs correctamente, tanto el firewall del servidor local como el del "router" bloqueen el tráfico. En el caso de que sea este último el culpable, para administrarlo se deberá entrar de nuevo en su panel de control web y buscar allí alguna sección llamada "DMZ" o similar. Indicando la IP del servidor local en dicha sección se consigue que a esa IP no le afecte la restricción del cortafuegos incluido en el "router".

DDNS

Sin embargo, todavía tenemos un problema: la gran mayoría de las veces, las IPs públicas asignadas por los operadores telefónicos a usuarios domésticos van cambiando de vez en cuando según sus criterios, por lo que la IP pública de nuestro router un día puede ser una y otro día puede ser otra. Esto quiere decir que los clientes deben saber en cada momento qué IP pública está asociada a nuestro servidor. Y esto no es demasiado práctico. Una manera sencilla de evitar esta complicación (sin tener que pagar el elevado coste de disponer de una IP pública fija) es utilizar un servicio de DNS dinámico (también llamado DDNS). La idea del DDNS es vincular la IP pública (y cambiante) de nuestro "router" con ufn nombre DNS elegido por nosotros, de manera que los dispositivos cliente no tengan que conocer nuestra IP pública concreta para conectarse sino que les baste con usar ese nombre DNS. La diferencia con un servicio DNS "estándar" es que con DDNS la vinculación entre IP pública y nombre DNS se actualiza automáticamente cada vez que dicha IP pública cambia, por lo que nuestra puerta de enlace tendrá siempre el mismo nombre accesible para todos los computadores del mundo independientemente de los cambios en su IP pública.

Existen muchas empresas que ofrecen servicios DDNS gratuitos. Algunas de ellas son (sin un orden en particular): GSLB (http://www.gslb.me), ChangeIP (https://www.changeip.com), Dynu (https://www.dynu.com), DuckDNS (https://www.duckdns.org), OpenDNS (http://www.opendns.com), DNSExit (http://www.dnsexit.com), Nsupdate (https://nsupdate.info) -es "open source"!-, Zonomi (http://www.zonomi.com), Hopper (https://www.hopper.pw), Afraid (http://freedns.afraid.org), DinaHosting (http://dinahosting.com) o No-Ip (http://www.noip.com), entre muchas otras. Una lista más completa se encuentra en https://dynamic.domains/dynamic-dns/providers-list/default.aspx

El mayor inconveniente de los servicios DDNS gratuitos es que los nombres DNS ofrecidos son bastante limitados y "feos". Si esto es un problema para nosotros, podemos considerar la opción (también gratuita) de obtener un nombre DNS completamente propio (eso sí, acabado en ".cf", ".tk", ".ga" o ".ml") gracias al servicio Freenom (http://freenom.com) para entonces redireccionarlo al nombre DDNS que hayamos definido en alguno de los servicios listados en el párrafo anterior o similar (ver "nota" siguiente). De todas maneras, lo más habitual es acabar contratando un servicio DNS -y/o DDNS!- de pago en los cuales podremos registrar nombres propios acabados en las terminaciones más habituales (".com", ".org",...); existen muchos (basta con escribir "registro dns" en Google para encontrar varios) pero podemos mencionar por ejemplo GoDaddy (https://es.godaddy.com), Domain.com (https://www.domain.com), EuroDNS (https://www.eurodns.com), CloudDNS (https://www.cloudns.net), Namecheap (https://www.namecheap.com), etc.

NOTA: En https://yunohost.org/it/dns_dynamicip se muestra un tutorial interesante que muestra cómo redireccionar un nombre DNS que hayamos registrado con alguna de las empresas anteriores (el cual será, como hemos dicho, estático pero bonito) a un nombre DDNS cualquiera de nuestra propiedad (el cual estará asociado a nuestra IP cambiante, que es lo que queremos, pero será feo)

Ejemplo de servicio DDNS: No-Ip

Como ejemplo concreto de configuración y uso de un servicio DDNS, a continuación mostraremos el procedimiento a seguir usando No-Ip. Para ello:

1.- Primero debemos crear una cuenta de usuario usando el formulario https://www.noip.com/sign-up y seguidamente debemos loguearnos con ella utilizando el enlace "Sign in" de su web principal.

2.- Una vez logueados, estaremos dentro de un panel de control online donde, entre otras, aparece la opción "Add Host", la cual nos llevará a un formulario donde podremos elegir el nombre DNS que queremos para nuestro router. A ese nombre se le añadirá siempre una "coletilla" propia de No-Ip, que también podemos elegir de entre varias que hay en un cuadro desplegable (por ejemplo, un nombre podría ser "miservidor.no-ip.org", o "miservidor.zapto.org", dependiendo de la coletilla seleccionada).

3.-En ese mismo formulario hay que asegurarse, finalmente, de que esté marcada la opción "DNS Host (A)" (que ya lo estará por defecto) y nada más. Por otro lado, la dirección IP mostrada en ese formulario se corresponderá con la IP pública detectada de nuestro router, pero la podemos especificar a mano si la conocemos y sabemos que la mostrada es incorrecta (aunque esto es poco probable).

4.- Una vez pulsado el botón de "Create host" del formulario, ya podremos salir del panel de control.

No obstante, queda un último paso para tener a nuestro "router" disponible en Internet con el nombre elegido: configurar este para que haga uso del servicio No-Ip. La mayoría de "routers" suelen ofrecer, en este sentido, una sección dentro de su panel de control llamada "DDNS" o similar donde se puede seleccionar la empresa cuyo servicio DDNS queremos utilizar (en este caso, No-Ip); allí deberemos, pues, marcar dicho servicio y, además, indicar el nombre de usuario No-Ip creado en los pasos anteriores (y su contraseña correspondiente). Una vez grabados estos cambios y reiniciado el "router", este procederá a establecer comunicación permanente con el servicio No-Ip para notificarle automáticamente cualquier eventual cambio detectado en su IP pública, (de manera que el servicio online pueda revincular el nombre del "Host" asociado al usuario No-Ip indicado en la configuración del "router" con la nueva IP detectada).

Desgraciadamente es posible que algún modelo de router no tenga en la lista de servicios DDNS configurables el servicio No-Ip en concreto; en ese caso, se puede optar por instalar en nuestro servidor un software específico (descargable de http://www.noip.com/download y disponible para Linux, Windows y OS X) para que sea él y no el "router" el encargado de comunicarse con No-Ip. De esta manera, se estaría delegando la tarea de monitoraje y actualización del vínculo IP pública<->nombre DDNS en el propio servidor local.

NOTA: Otra opción alternativa es intentar sobrescribir el firmware del "router" que viene de fábrica por otro que sí soporte No-IP, como por ejemplo OpenWRT (https://openwrt.org) o DD-WRT (http://www.dd-wrt.com), pero esta última solución solo se recomienda para usuarios que realmente sepan lo que están haciendo, ya que se puede acabar con un "router" inservible. Un ejemplo de configuración puede ser https://alexskra.com/blog/dynamc-dnsddns-with-openwrt-and-cloudflare
NOTA: En el caso de optar por instalar un software específico en nuestro servidor para monitorizar y actualizar el vínculo IP pública <-> nombre DDNS (el cual se suele llamar "cliente DDNS"), normalmente el proveedor DDNS ofrece su propio software particular (tal como se ha indicado en el párrafo anterior para el caso de No-Ip). No obstante, también existen clientes DDNS genéricos que son capaces de trabajar con varios proveedores DDNS de forma indistinta, previa configuración (en la cual habrá que indicar, además del usuario y contraseña pertinente, el proveedor concreto con el que se establecerá comunicación porque cada uno suele funcionar mediante un protocolo propio diferente, aunque muchos se basan simplemente en ofrecer una API REST, por lo que, en realidad, usando *curl* ya bastaría). Ejemplos de estos clientes DDNS genéricos son "**Ddclient**" (https://github.com/ddclient/ddclient) , "**Inadyn**" (https://troglobit.com/projects/inadyn) o "**Dyndnsc**" (https://github.com/infothrill/python-dyndnsc). Se puede encontrar una lista de más clientes, tanto genéricos como particulares para un servicio concreto, en https://wiki.archlinux.org/index.php/Dynamic_DNS#Update_clients

Hay que tener en cuenta que los servicios gratuitos de No-IP caducan en caso de inactividad: si no se accede al nombre DNS en el plazo de 30 días, el dominio es borrado. Es posible evitar que esto ocurra haciendo clic sobre un enlace en un e-mail de aviso que es enviado tras 25 días de inactividad, o también comprando un servicio No-IP de pago.

Alternativa al DDNS: "LocalTunnel" y programas similares

Si no queremos registrarnos en ningún servicio DDNS (o no podemos porque nuestra IP pública está compartida con varios vecinos que tienen contratado el mismo ISP, cosa bastante habitual actualmente) aún podemos emplear una última alternativa para hacer público nuestro servidor local en Internet de forma gratuita que consiste en instalar un determinado software específico en nuestro servidor local que se conecte a un servidor remoto de Internet, el cual asociará (de forma gratuita, en principio) un nombre DNS para nuestro servidor y hará de "proxy" de todas las conexiones para que le lleguen a él. Ejemplo de este tipo de software son, entre otros: Pagekite (https://pagekite.net), Ngrok (https://ngrok.com), Portmap (https://portmap.io), LocalTunnel (https://github.com/localtunnel/localtunnel) o TunnelTo (https://tunnelto.dev).

NOTA: Una limitación importante de este tipo de software (aunque dependerá mucho del elegido en concreto) es que nuestro servidor local solamente puede ser de tipo HTTP/S (o también SSH) porque no contemplan otro tipo de servidores

Por ejemplo, una vez instalado LocalTunnel en nuestro servidor web escuchando en el puerto 80, ejecutando el comando *lt --port 80* , nuestro servidor se conectará con http://localtunnel.me para obtener un nombre DDNS gratuito válido y establecer un "túnel" de conexiones a través del cual se transmitirá todo el tráfico entre uno y otro (todo ello mientras dicho comando *lt* esté en marcha; muy útil por ejemplo para tener nombres públicos "de usar y tirar").

EXERCICIS:

1.-a) ¿Què significa "IaaS"? ¿I "DNAT"? ¿I "DDNS"? (ho tens explicat a la teoria anterior)

2.-a) Crea un compte d'usuari al servei gratuit NoIP (i associa'l a un determinat domini DDNS a escollir en el seu panell de control web), tal com s'explica als paràgrafs anteriors.

b) Fes un ping des de la màquina real al nom DDNS escollit i fixa't en la IP que està responent. ¿És la mateixa que obtens en fer *curl ipinfo.io/ip*? ¿Què significa això, doncs? D'altra banda, si obres un navegador a la màquina real i escrius a la barra de direccions el nom DDNS escollir, ¿què veus i per què?

c) Arrenca una màquina virtual (Ubuntu o Fedora, és igual) amb la tarja de xarxa en mode NAT o adaptador pont (és igual) i instal.la-hi el paquet "ddclient". Seguidament escriu les línies següents al final del fitxer "/etc/ddclient.conf" per vincular-lo al compte d'usuari recentment creat...:

use=web
ssl=yes
protocol=noip
login=<USERNAME>
password=<PASSWORD>
<YOUR_HOSTNAME>

NOTA: El valor "web" de la linia *use=* significa que per defecte es comprovarà (de forma periòdica, amb un interval indicat en segons a la línia *daemon=*) el valor de la IP pública mitjançant el servei "checkip.dyndns.org". Si es vol utilitzar un altre servei, es pot indicar a la mateixa línia *use=*, així: *use=web,web=myonlineportal.net/checkip* En qualsevol cas, es recomana llegir els comentaris explicatius presents al fitxer "ddclient.conf"

...i tot seguit executa el programa *ddclient* en forma de dimoni en segon pla (així realitzarà les actualitzacions DDNS en segon pla quan sigui necessari de forma "desatesa"), simplement fent així: *sudo systemctl start ddclient*

NOTA: Òbviament, si volguéssim usar el client *ddclient* més enllà d'aquest exercici puntual, el més còmode serà que fem *sudo systemctl enable ddclient*

d) Vés al panell de control web del teu usuari No-IP i sota l'apartat "Dynamic DNS" fixa't en el valor de la columna "Last update" que hi apareix. ¿Què li ha passat?

e) ¿Per què, tot i tenir el client *ddclient* en marxa, la nostra màquina no es troba accessible encara des d'Internet?

NOTA: Un tutorial que explica més o menys el vist en aquest exercici però aplicat al cas d'un servidor en forma de Raspberry Pi és https://engineerworkshop.com/blog/connecting-your-raspberry-pi-web-server-to-the-internet/

3.-a) ¿Què explica aquesta web: https://www.noip.com/integrate/ip-detection (o, de forma similar, aquesta altra: https://help.dyn.com/remote-access-api/detect-ip-change)?

b) ¿Què explica aquesta web: https://www.noip.com/integrate/request (o, de forma similar, aquesta altra: https://help.dyn.com/remote-access-api/perform-update) ? Prova de fer alguna de les consultes mostrades a la primera url amb l'eina *curl* i observa què n'obtens.

c) ¿Què explica aquesta web: https://www.noip.com/integrate/response (o, de forma similar, aquesta altra: https://help.dyn.com/remote-access-api/return-codes) ?

d) ¿Per a què serveix aquest programa: https://git.zx2c4.com/zx2c4-ddns/about ?

4.-a) Arrenca una màquina virtual que tingui una tarja de xarxa en mode "adaptador pont", instal.la el paquet "pagekite" (executant aquesta comanda*: curl -s https://pagekite.net/pk | sudo bash)* i també el servidor web Apache (executant *sudo apt install apache2* o *sudo dnf install httpd*, segons la distribució). Posa en marxa aquest darrer (recorda: *sudo systemctl start apache2* o *sudo systemctl start httpd*, segons la distribució)

b) Executa la comanda *pagekite --signup* A partir d'aquí:

* Se't preguntarà si vols utilitzar el servei gratuït PageKite.net. Com que no tens cap servidor PageKite propi (és de codi obert: podries tenir el teu servidor públic "proxy" propi!), tria **Y**.

NOTA: Es pot llegir com muntar un servidor PageKite propi (sobre l'IaaS DigitalOcean i una màquina Debian) a https://www.digitalocean.com/community/tutorials/how-to-set-up-a-pagekite-front-end-server-on-debian-9

* A continuació, se't demanarà la teva adreça de correu electrònic per tal de registrar-te al servei PageKite.net i així obtenir-ne un compte d'usuari (gratuït)

* A continuació se't demanarà indicar el nom del "pagekite" que voldràs associar al teu servidor local. El nom pot ser el que vulguis (suposarem "elteunom"), però a aquest se l'afegirà automàticament el domini "pagekite.me", de forma que el nom complet serà "elteunom.pagekite.me"

* Les dades del teu compte (inclosa la contrasenya) s'enviaran a la teva adreça de correu electrònic. Deixa a mitges l'assistent de terminal i vés a la teva bústia de correu electrònic per fer clic a l'enllaç d'activació de compte indicat en el missatge que hauràs rebut de PageKite. Accediràs llavors a un tauler de control web on, si vols, podràs canviar el nom de domini escollit per al teu servidor local, la contrasenya del teu compte i molt més.

* Torna a l'assistent de terminal i tria **Y** per continuar i desa la configuració en un fitxer personal. Observa com la comanda *pagekite* es connectarà al servidor públic de Pagekite per començar a "publicar" la presència de la teva màquina local a Internet. Atura aquesta comanda *pagekite* (amb CTRL+C) perquè la posarem en marxa "a mà" a l'apartat següent (que és com ho haurem de fer a partir d'ara).

c) Executa la comanda *pagekite elteunom.pagekite.me*

NOTA: Per executar *pagekite* en segon pla, es pot afegir el paràmetre *--daemonize* En aquest cas, per matar el procés es podria fer, per exemple *pkill pagekite*

d) Obre un navegador de la teva màquina real (o el d'algun company) i escriu la direcció *http://elteunom.pagekite.me* ¿Què veus?

NOTA: A https://www.linuxbabe.com/linux-server/expose-localhost-to-internet-pagekite-ubuntu es troba un tutorial molt interessant sobre com posar en marxa un servidor Pagekite en un IaaS propi (amb un domini propi també)

e) ¿Què explica aquest tutorial oficial: https://pagekite.net/wiki/Howto/SshOverPageKite ? ¿Per què, tal com allí s'exposa, cal modificar la configuració del client SSH?